

Using a Hugging Face Model with Llama.CPP

Docs

<https://github.com/ggerganov/llama.cpp>

PYTHON WRAPPER (my holy grail for docs now):

<https://github.com/abetlen/llama-cpp-python>

<https://llama-cpp-python.readthedocs.io/en/latest/>

Listen these LLM folk are more MLM , jk. But you do need a GGUF file.

This video kind of helped a lot?

Helped piece the idea that I really need to quantize it or get a gguf file.

<https://www.youtube.com/watch?v=jOEu0PE4ozM>

step 1 <https://github.com/abetlen/llama-cpp-python?tab=readme-ov-file#pulling-models-from-hugging-face-hub>

Pull this stuff,

step 1.1:

https://huggingface.co/docs/huggingface_hub/en/guides/cli

LOGIN AND STUFF

https://huggingface.co/docs/huggingface_hub/en/guides/cli#huggingface-cli-login

download the model

https://huggingface.co/docs/huggingface_hub/en/guides/cli#huggingface-cli-download

I downloaded the whole thing (REFERENCE VIDEO AT THE TOP, WAS HELPFUL ALSO A LOT OF SOURCES TELL YOU IT'S USUALLY FOUND IN THE ~CACHE)

<https://huggingface.co/meta-llama/Meta-Llama-3-8B>

Once you have this download you can move onto llama.cpp

video helps guide one to getting to this file if you need it

step 2 <https://github.com/ggerganov/llama.cpp?tab=readme-ov-file#build>

Here I made a new dir

mkdir llamaCPP

git clone > clone repo

cd into repo,

<https://github.com/ggerganov/llama.cpp?tab=readme-ov-file#cuda>

```
$ https://github.com/ggerganov/llama.cpp?tab=readme-ov-file#cuda
```

after was able to run stuff here: [step3](#)

```
# obtain the official LLaMA model weights and place them in ./models
ls ./models
llama-2-7b tokenizer_checklist.chk tokenizer.model
# [Optional] for models using BPE tokenizers
ls ./models
<folder containing weights and tokenizer json> vocab.json
# [Optional] for PyTorch .bin models like Mistral-7B
ls ./models
<folder containing weights and tokenizer json>

# install Python dependencies
python3 -m pip install -r requirements.txt

# convert the model to ggml FP16 format
python3 convert-hf-to-gguf.py models/mymodel/

# quantize the model to 4-bits (using Q4_K_M method)
./llama-quantize ./models/mymodel/ggml-model-f16.gguf ./models/mymodel/ggml-model-
Q4_K_M.gguf Q4_K_M

# update the gguf filetype to current version if older version is now unsupported
./llama-quantize ./models/mymodel/ggml-model-Q4_K_M.gguf ./models/mymodel/ggml-
model-Q4_K_M-v2.gguf COPY
```

Step 3 <https://github.com/ggerganov/llama.cpp?tab=readme-ov-file#prepare-and-quantize>

Quantize the model,

DYOR

all random stuff i've looked up

Not bad resources

<https://medium.com/@ingridwickstevens/quantization-of-llms-with-llama-cpp-9bbf59deda35>

[Like I didn't do this, there's a lot of research, don't overwhelm yourself](#)

<https://github.com/ggerganov/llama.cpp/issues/1344>

my specs,

[https://www.google.com/search?](https://www.google.com/search?q=can+i+run+llama+3+8b+with+a+3080+site:www.reddit.com&sca_esv=ba8d56c099ffe98e&sxsrf=ADLYWIKY-tjMwps6h-XzWzleX0won6-28g:1718280528125&sa=X&ved=2ahUKEwjokIPkxdiGAxVDSzABHctnCy4QrQloBHoECCYQBQ&biw=1920&bih=919&dpr=1#ip=1)

[q=can+i+run+llama+3+8b+with+a+3080+site:www.reddit.com&sca_esv=ba8d56c099ffe98e&sxsrf=ADLYWIKY-tjMwps6h-XzWzleX0won6-28g:1718280528125&sa=X&ved=2ahUKEwjokIPkxdiGAxVDSzABHctnCy4QrQloBHoECCYQBQ&biw=1920&bih=919&dpr=1#ip=1](https://www.google.com/search?q=can+i+run+llama+3+8b+with+a+3080+site:www.reddit.com&sca_esv=ba8d56c099ffe98e&sxsrf=ADLYWIKY-tjMwps6h-XzWzleX0won6-28g:1718280528125&sa=X&ved=2ahUKEwjokIPkxdiGAxVDSzABHctnCy4QrQloBHoECCYQBQ&biw=1920&bih=919&dpr=1#ip=1)

read about q6_k

[https://www.google.com/search?](https://www.google.com/search?q=quantize+the+model+to+Q6_K&sca_esv=ba8d56c099ffe98e&sxsrf=ADLYWILi4Ogav6biasPG7wlhTrkWVYaszw%3A1718280380246&ei=vOBqZr3cDq_qkvQPX_qL6Ag&ved=0ahUKEwi9qsGdxdiGAxUvtYQIHUf9Ao0Q4dUDCBA&uact=5&oq=quantize+the+model+to+Q6_K&gs_lp=Egxnd3Mtd2l6LXNlcniAIGnF1YW50aXplIHRob2ZSBtb2RlbyB0byBRNl9LMgUQIRigATIFECEYoAEyBRAhGKABMgUQIRigATIFECEYoAFItcUBUJYCWJXBAXAAeAKQAQCYAWigAcsDqgEDNC4xuAEDyAEA-AEB-AECmAlGoAlaA8ICBBAAGEfCAgUQABiABMICBhAAGBYHsICCxAAGIAEGlYDGloFwglIEAAYgAQYogTCAggQABgWGAoYHpgDAIgGAZAGCJIHAzUuMaAHqB8&scient=gws-wiz-serp)

[q=quantize+the+model+to+Q6_K&sca_esv=ba8d56c099ffe98e&sxsrf=ADLYWILi4Ogav6biasPG7wlhTrkWVYaszw%3A1718280380246&ei=vOBqZr3cDq_qkvQPX_qL6Ag&ved=0ahUKEwi9qsGdxdiGAxUvtYQIHUf9Ao0Q4dUDCBA&uact=5&oq=quantize+the+model+to+Q6_K&gs_lp=Egxnd3Mtd2l6LXNlcniAIGnF1YW50aXplIHRob2ZSBtb2RlbyB0byB0byBRNl9LMgUQIRigATIFECEYoAEyBRAhGKABMgUQIRigATIFECEYoAFItcUBUJYCWJXBAXAAeAKQAQCYAWigAcsDqgEDNC4xuAEDyAEA-AEB-AECmAlGoAlaA8ICBBAAGEfCAgUQABiABMICBhAAGBYHsICCxAAGIAEGlYDGloFwglIEAAYgAQYogTCAggQABgWGAoYHpgDAIgGAZAGCJIHAzUuMaAHqB8&scient=gws-wiz-serp](https://www.google.com/search?q=quantize+the+model+to+Q6_K&sca_esv=ba8d56c099ffe98e&sxsrf=ADLYWILi4Ogav6biasPG7wlhTrkWVYaszw%3A1718280380246&ei=vOBqZr3cDq_qkvQPX_qL6Ag&ved=0ahUKEwi9qsGdxdiGAxUvtYQIHUf9Ao0Q4dUDCBA&uact=5&oq=quantize+the+model+to+Q6_K&gs_lp=Egxnd3Mtd2l6LXNlcniAIGnF1YW50aXplIHRob2ZSBtb2RlbyB0byB0byBRNl9LMgUQIRigATIFECEYoAEyBRAhGKABMgUQIRigATIFECEYoAFItcUBUJYCWJXBAXAAeAKQAQCYAWigAcsDqgEDNC4xuAEDyAEA-AEB-AECmAlGoAlaA8ICBBAAGEfCAgUQABiABMICBhAAGBYHsICCxAAGIAEGlYDGloFwglIEAAYgAQYogTCAggQABgWGAoYHpgDAIgGAZAGCJIHAzUuMaAHqB8&scient=gws-wiz-serp)

reddits

1. https://www.reddit.com/r/LocalLLaMA/comments/183ie9t/ctransformers_vs_llamacpppytho_n_which_one_should/
2. https://www.reddit.com/r/LocalLLaMA/comments/1c8u0n5/thanks_zuckmeta_for_these_great_llama_3_models_3/
3. https://www.reddit.com/r/LocalLLaMA/comments/1aeflyu/upgrading_gtx_1060_6gb_to_rtx_3070_ti_8gb_is_good/

4. https://www.reddit.com/r/LocalLLaMA/comments/148geaj/utilize_my_current_hardware_or_upgrade/
5. https://www.reddit.com/r/LocalLLaMA/comments/1c9v4u3/llama_3_8b_any_way_to_run_this_on_rtx_3080/
6. USE Q6_K WHENEVER?
https://www.reddit.com/r/LocalLLaMA/comments/1c8mvmc/llama_3_discussion_about_quantization_performance/
7. https://www.reddit.com/r/LocalLLaMA/comments/1cci5w6/quantizing_llama_3_8b_seems_more_harmful_compared/
8. https://www.reddit.com/r/LocalLLaMA/comments/144uc0l/damn_i_was_so_satisfied_with_my_3080_with_10gb_of/
9. https://www.reddit.com/r/SillyTavernAI/comments/1cu4dd6/can_anyone_recommend_some_local_models_for_3080/
10. https://www.reddit.com/r/LocalLLaMA/comments/1aezi29/difference_between_the_different_python_libraries/
11. https://www.reddit.com/r/LocalLLaMA/comments/1cetn9z/quantization_seems_to_hurt_the_quality_of_llama_3/
12. https://www.reddit.com/r/LocalLLaMA/comments/1c9qufe/note_on_llama_3_quantized_models/

https://huggingface.co/docs/huggingface_hub/en/guides/cli

ITS RUNNING ON MY CPU

TO CUDA WE GO

```
pip install llama-cpp-python --extra-index-url https://abetlen.github.io/llama-cpp-python/whl/cu124 --upgrade --force-reinstall --no-cache-dir
```

useful commands

<https://stackoverflow.com/questions/78415856/detecting-gpu-availability-in-llama-cpp-python>

watch ur nvidia card

```
watch -n 1 nvidia-smi
```

takes a screenshot:

```
nvidia-smi
```

check cuda version

```
nvcc --version
```

commands 1 by 1

[https://developer.nvidia.com/cuda-12-4-0-download-archive?
target_os=Linux&target_arch=x86_64&Distribution=WSL-
Ubuntu&target_version=2.0&target_type=deb_local](https://developer.nvidia.com/cuda-12-4-0-download-archive?target_os=Linux&target_arch=x86_64&Distribution=WSL-Ubuntu&target_version=2.0&target_type=deb_local)

[https://developer.nvidia.com/cuda-12-4-0-download-archive?
target_os=Linux&target_arch=x86_64&Distribution=Ubuntu&target_version=20.04&target_type
=deb_local](https://developer.nvidia.com/cuda-12-4-0-download-archive?target_os=Linux&target_arch=x86_64&Distribution=Ubuntu&target_version=20.04&target_type=deb_local)

[https://www.reddit.com/r/KoboldAI/comments/16op2jv/can_someone_eli5_how_to_calculate_th
e_number_of/](https://www.reddit.com/r/KoboldAI/comments/16op2jv/can_someone_eli5_how_to_calculate_the_number_of/)

<https://docs.nvidia.com/cuda/cuda-installation-guide-linux/index.html#wsl>

<https://docs.nvidia.com/cuda/cuda-installation-guide-linux/index.html#post-installation-actions>

```
CMAKE_ARGS="-DLLAMA_CUDA=on" pip install llama-cpp-python
```

```
CMAKE_ARGS="-DLLAMA_BLAS=ON -DLLAMA_BLAS_VENDOR=OpenBLAS" \  
pip install llama-cpp-python
```

FINAL COMMAND

```
pip install llama-cpp-python --extra-index-url https://abetlen.github.io/llama-cpp-  
python/whl/cu124 --upgrade --force-reinstall --no-cache-dir
```

<https://github.com/ggerganov/llama.cpp/issues/6898>

checking for cuda existence

```
mako77@DESKTOP-UMFPB6H:~$ ls /usr/local/cuda
DOCS      README  compute-sanitizer  extras  include  libnvvp      nvml  share
targets  version.json
EULA.txt  bin      doc              gds     lib64     nsightee_plugins  nvvm  src
tools

mako77@DESKTOP-UMFPB6H:~$ ls /usr/local/
bin  cuda  cuda-12  cuda-12.4  cuda-12.5  etc  games  include  lib  man  sbin
share  src

mako77@DESKTOP-UMFPB6H:~$ export CUDA_HOME=/usr/local/cuda-12.4
mako77@DESKTOP-UMFPB6H:~$ which nvcc
/usr/local/cuda-12.4/bin/nvcc
mako77@DESKTOP-UMFPB6H:~$ echo $CUDA_HOME
/usr/local/cuda-12.4
```

SETTING UP LLAMA CPP

HANDLING THE CPU BEING 100%

YOU NEED TO PAY ATTENTION TO ENVIRONMENT VARIABLES, LEARN TO PROPERLY USE THEM, SET THEM, FIND THEM, ETC.

I do believe this is one of the things that saved me

```
CMAKE_ARGS="-DLLAMA_CUBLAS=on" pip install llama-cpp-python --force-reinstall --
upgrade --no-cache-dir
```

```
# Linux and Mac
CMAKE_ARGS="-DLLAMA_BLAS=ON -DLLAMA_BLAS_VENDOR=OpenBLAS" \
pip install llama-cpp-python
```

Resources for CPU BEING 100%

(1) Rebuilding, at first I did not build with cuda home or paths and stuff set.

<https://github.com/abetlen/llama-cpp-python/issues/509#issuecomment-1655864538>



glaudiston commented on Jul 28, 2023 • edited ▾

To build the `libllama.so` with gpu support you need to have CUDA SDK installed, then:

```
git clone https://github.com/ggerganov/llama.cpp
cd llama.cpp
export CUDA_HOME=/your/cuda/home/path/here
export PATH=${CUDA_HOME}/bin:$PATH
export LLAMA_CUBLAS=on
make clean
make libllama.so
```

Then note that the `g++` compiler will add the `-DGGML_USE_CUBLAS` compiler flag.
and it will create a file called `libllama.so` in the current directory.
check it with

```
ls -l libllama.so
```

After that you can force `llama-cpp-python` to use that lib with:

```
export LLAMA_CPP_LIB=/path/to/your/libllama.so
```

After that, it worked with GPU support here. Of course you have to init your model with something like

```
llm = Llama(
    ...
    n_gpu_layers=20,
    ...
)
```

Hope it helps.



18

After doing all that,

ALSO MAKING SURE I INSTALLED CUDA, BECAUSE I DID NOT KNOW I WOULD NEED CUDA LOL

I re-quantized the model after this, meaning I deleted the previous one and did it with these conditions fulfilled.

After re-quantizing

```
# Linux and Mac
CMAKE_ARGS="-DLLAMA_BLAS=ON -DLLAMA_BLAS_VENDOR=OpenBLAS" \
pip install llama-cpp-python
```

I guess it did say to do this prior,

The official docs of this wrapper suggest this:

Upgrading and Reinstalling

To upgrade and rebuild `llama-cpp-python` add `--upgrade --force-reinstall --no-cache-dir` flags to the `pip install` command to ensure the package is rebuilt from source.

Which was almost right,

but ultimately what worked for me was plainly making sure I pip uninstalled the package then ran

```
CMAKE_ARGS="-DLLAMA_CUBLAS=on" pip install llama-cpp-python --force-reinstall --upgrade --no-cache-dir
```

Links that helped ---

found solution here

<https://github.com/abetlen/llama-cpp-python/issues/576#issuecomment-1766003289>

<https://github.com/abetlen/llama-cpp-python/issues/576>

huge helped me with the idea to rebuild everything. When in doubt just rebuild

<https://stackoverflow.com/questions/78415856/detecting-gpu-availability-in-llama-cpp-python>

Definitely need to set env variables first i imagine

I did do this also when running the build

<https://github.com/ggerganov/llama.cpp?tab=readme-ov-file#cuda>

really feel like it didnt do much, but who knows i did it so its included

this is where i found the command

<https://github.com/ggerganov/llama.cpp/issues/6360>